Análise e Visualização de Dados 2023/2024

Diário de Bordo de 14 de março de 2024 (quinta-feira)

Realizado por Renata Martins - PG52896

Sumário

- Diretrizes para avaliação dos alunos

Durante a primeira parte da aula, o professor Álvaro apresentou o ficheiro "Contributos para o enunciado do trabalho final de AVD + PI2-sombra" disponível na Blackboard.
O documento descreve as diretrizes para o trabalho final das disciplinas de Análise e Visualização de Dados e Projeto Integrado 2, que incide sobre a análise de entrevistas. Os alunos devem realizar várias tarefas de processamento de texto e análise de dados, como extração de entidades nomeadas, lematização, análise de sentimentos, entre outras, usando ferramentas como expressões regulares, SpaCy, Python e Excel. Os trabalhos finais incluem a geração de um PDF e a proposta de uma "História (Alternativa) da Universidade" baseada nos dados analisados. São dadas orientações específicas sobre a estrutura do relatório, a redação, a citação de fontes e a apresentação dos resultados.
Atenção! É necessário fazer uma retificação ao documento: o trabalho será em grupo e o relatório também. Portanto, onde se lê "*Apresentação de relatório individual*", considerar "*Apresentação de relatório do grupo*".

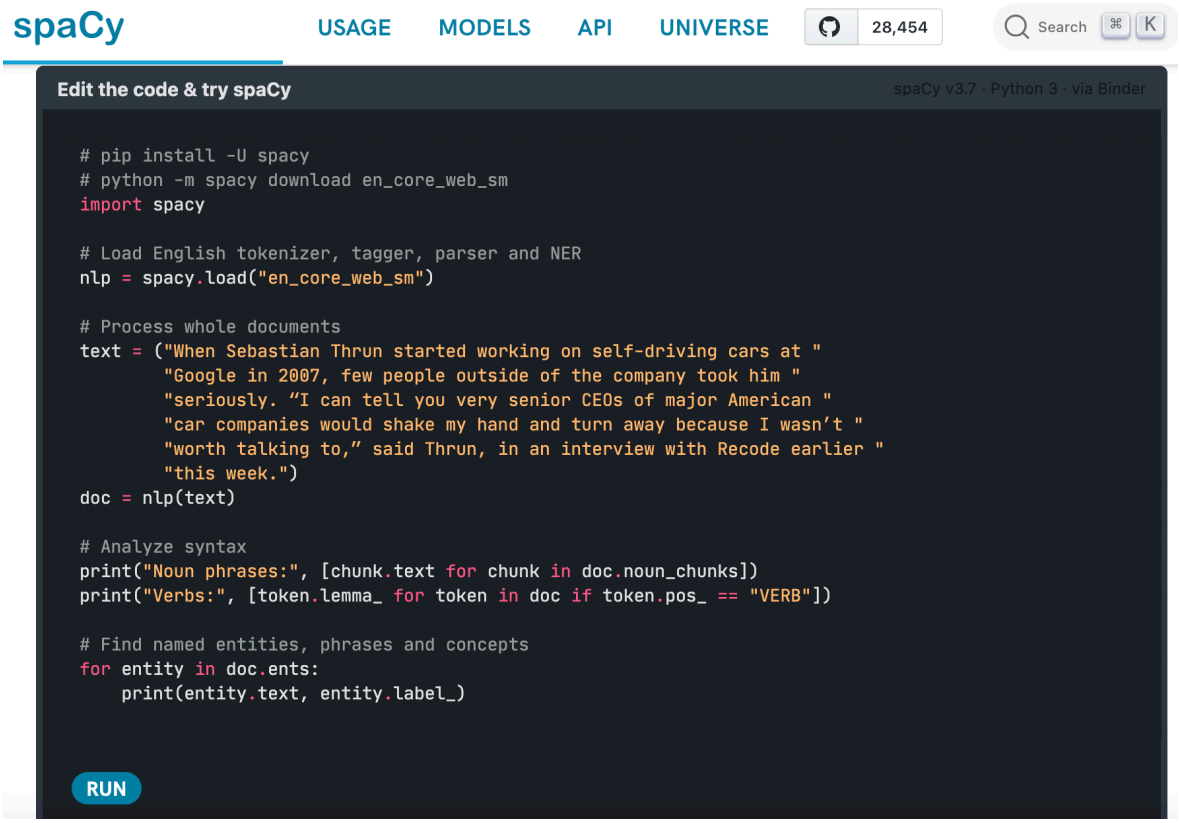
- SpaCy

O professor JJ apresentou a SpaCy - uma biblioteca de software livre e de código aberto, projetada para trabalhar com processamento de linguagem natural.
O que essa biblioteca faz? Ajuda a realizar tarefas com texto, como identificar partes do discurso em uma frase (por exemplo, distinguir substantivos de verbos), reconhecer nomes de pessoas ou lugares.

Prática:

1 - Aceder ao Spacy

Vá até o sítio da SpaCy no link: https://spacy.io, copie o código disponível na página principal e cole em novo documento .py para editar os códigos.



2 - Instalar o spacy no terminal

pip install -U spacy

No Mac costuma dar jeito com: python3 -m pip install spacy

3 - Importar a biblioteca SpaCy para o seu script ou ambiente de trabalho em Python

```
import spacy
```

4 - Fazer download do modelo de linguagem para o português

Denominado **pt_core_news_lg**. Este modelo é necessário para que a biblioteca entenda a linguagem e possa realizar tarefas, como tokenização, análise

morfológica, reconhecimento de entidades nomeadas, etc.

Para isso, no substitua en_core_web_sm (modelo para inglês) por pt_core_news_lg

```
# Load Portuguese tokenizer, tagger, parser and NER
nlp = spacy.load("pt_core_news_lg")
```

5 - Processar um texto em português

```
# Process whole documents
text = """Bom dia, Alvaro Iriarte da Silva. Onde é que nasceu?
Em Viana do Castelo."""
doc = nlp(text)
```

Aqui é onde a mágica acontece. Ao chamar nlp(text), você está passando a string text para o modelo de NLP para processá-la.

6 - Fazer a análise sintática

```
# Analyze syntax
# print("Noun phrases:", [chunk.text for chunk in
doc.noun_chunks])
```

Esta linha imprime as frases nominais (grupos de palavras que funcionam como substantivos). doc.noun_chunks gera uma sequência de tais frases no documento.

```
# print("Verbs:", [token.lemma_ for token in doc if token.pos_
== "VERB"])
```

Aqui, o código vai extrair e imprimir os verbos do texto. Ele verifica se a classe gramatical (pos_) é "VERB". Se for, ele pega o lema do verbo (lemma_), que é a forma base da palavra (por exemplo, “nascer” para “nasceu").

7 - Encontrar entidades nomeadas, frases e conceitos

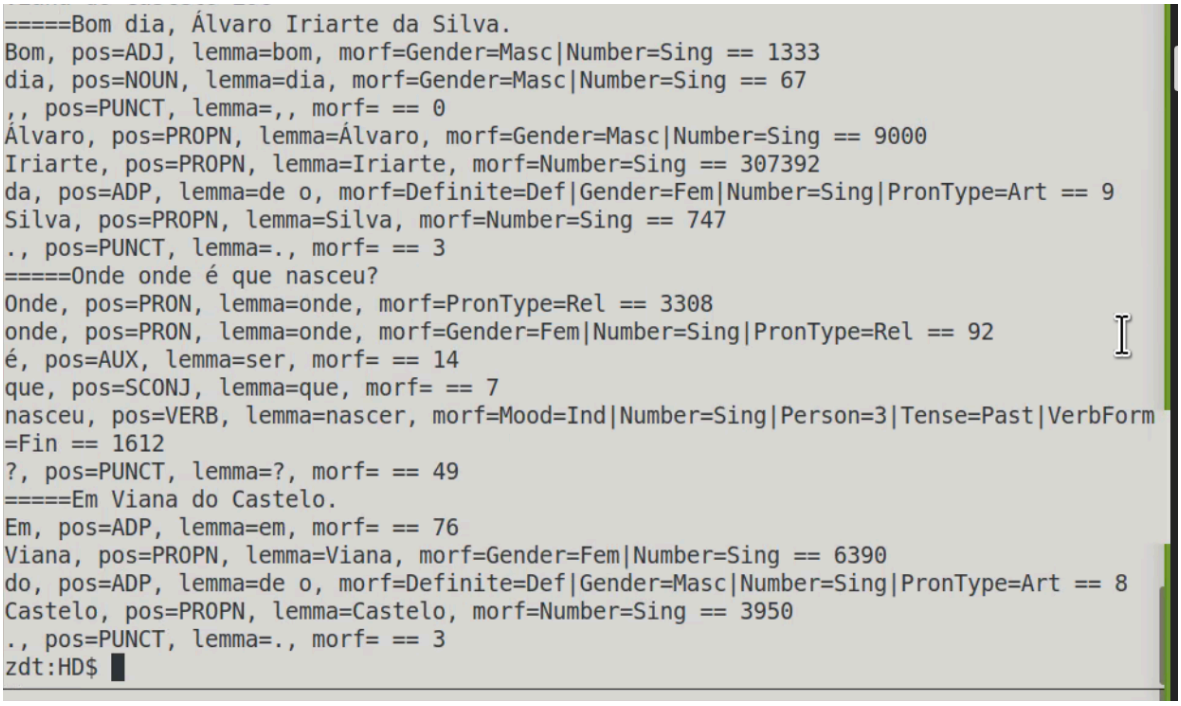
```
# Find named entities, phrases and concepts
for ent in doc.ents:
    print(ent.text, ent.label_)
```

O comando vai percorrer todas as entidades nomeadas (nomes de pessoas, lugares, organizações, datas) detectadas no texto.
Ao dar print, para cada entidade, o código imprime o texto da entidade (ent.text) e seu rótulo (ent.label_), que indica o tipo da entidade (por exemplo, "PESSOA", "LOCAL").

8 - Extrair informações detalhadas sobre cada palavra

```
for fr in doc.sents:
    print(f"====={fr.text}")
    for pal in fr:
        # print(f"{pal.text}, pos={pal.pos}, lemma={pal.morph}
== {pal.rank}")
        print(f"{pal.text}, pos={pal.pos},
lemma={pal.lemma_}")
print(f"{pal.text}\t{pal.pos_}\t{pal.lemma_}\t{pal.rank}")
```

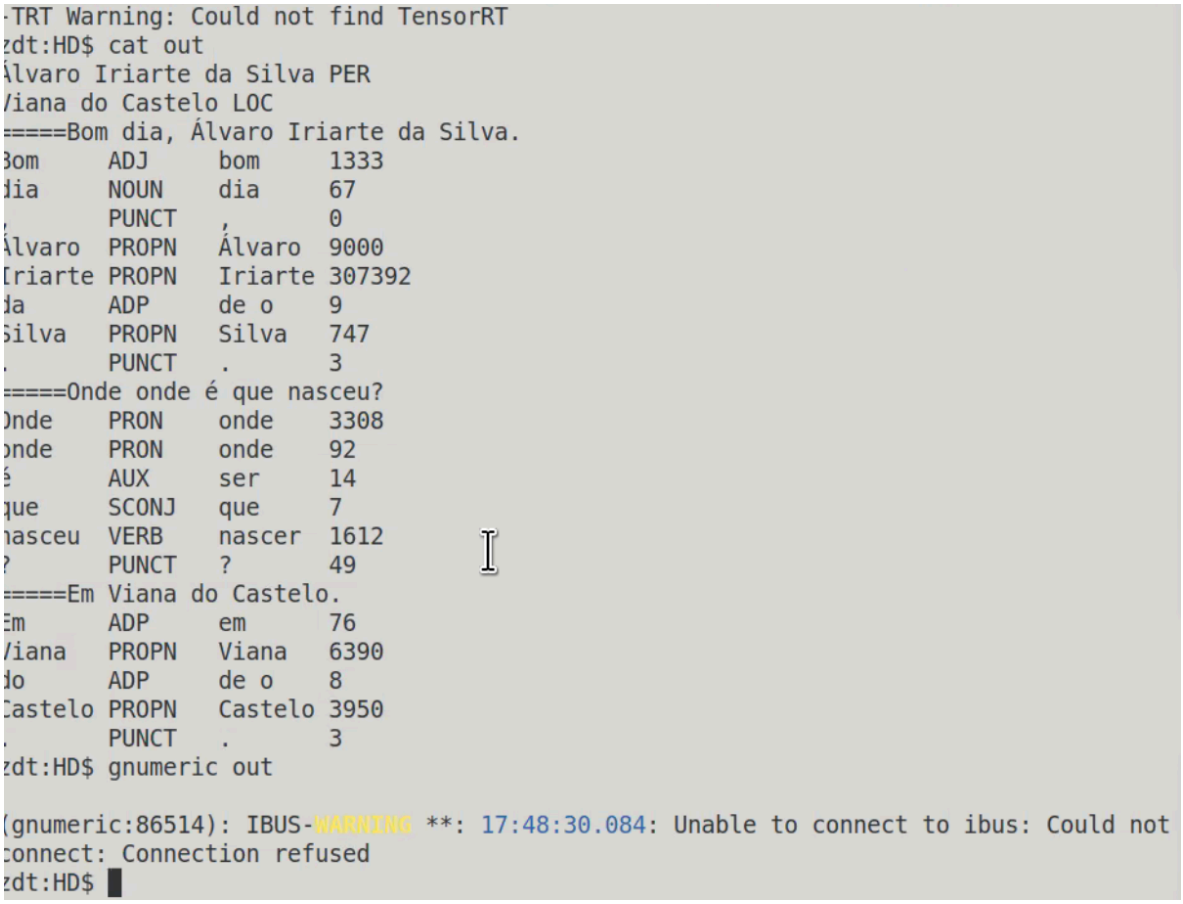
Para cada palavra, o comando imprime o texto da palavra, sua classe gramatical (pos_), a forma lematizada (lemma_), e o ranking (rank), que pode ser usado para determinar a frequência da palavra na biblioteca.



Para organizar as saídas em colunas, o professor usou a tabulação \t para separar as informações, o que alinhou os textos verticalmente.

```
print(f"{pal.text}\t{pal.pos_}\t{pal.lemma_}\t{pal.rank}")
```

Resultado gerado:



Quadro final dos comandos da aula:

```python
# pip install -U spacy
# python -m spacy download pt_core_news_lg (português)
# en_core_web_sm (inglês)
import spacy

# Load English tokenizer, tagger (associar categoria morfosintatica), parser (calcular a árvore sintática) and NER (entidades).
nlp = spacy.load("pt_core_news_lg")

# Process whole documents
text = ("Bom dia, Alvaro Iriarte da Silva. Onde é que nasceu? Em Viana do Castelo.")
doc = nlp(text)

# Analyze syntax
# print("Noun phrases:", [chunk.text for chunk in doc.noun_chunks])
# print("Verbs:", [token.lemma_ for token in doc if token.pos_ == "VERB"])

# Find named entities, phrases and concepts
for ent in doc.ents:
    print(ent.text, ent.label_)

for fr in doc.sents:
    print(f"===={fr.text}")
    for pal in fr:
        # print(f"{pal.text}, pos={pal.pos}, lemma={pal.morph} == {pal.rank}")
        print(f"{pal.text}\t{pal.pos_}\t{pal.lemma_}\t{pal.rank}")


```